\* --------------------------------------------------------------------------------------------------

\* Name:

\* GetServerHandler : Get a server handler to control EZSocketcore's server function

\*

\* Synopsis:

\* #include "EZSocketCore.h"

\* struct EZSocketCore \* GetServerHandler(int port,int mode,(void\*)function,int \*errorcode);

\*

\* Description:

\* First parameter is the port which you want to bind.

\* EZSocketCore support two mode on the Server program.

\* You should pass ServerMainLoop\_EZUserdef or ServerMainLoop\_EZWeb (define in EZSocketCore.h) to

\* second parameter and pass a function pointer to third parameter.

\* You should pass a pointer which point to an integer to lastest parameter , when error occur,

\* GetServerHandler set errorcode and return NULL.

\* Read example code to learn more about it.

\*

\* Return Value

\* On success, GetServerHandler return a Handler (a pointer to an EZSocketCore struct) to control

\* EZSocketcore server , Otherwise, NULL is returned and error code is set .

\*

\* --------------------------------------------------------------------------------------------------

\* Name:

\* GetServerErrorMsg : Get the error message when GetServerHandler return NULL

\*

\* Synopsis:

\* #include "EZSocketCore.h"

\* GetServerErrorMsg(int errorcode,char \*message,int maxlength)

\*

\* Description:

\* When GetServerHandler return NULL , you can call GetServerErrorMsg to get error message .

\* First parameter is the error code set by GetServerHandler , second parameter is a string pointer.

\* Third parameter is the max length of the sring.

\* --------------------------------------------------------------------------------------------------

\* NAME:

\* GetClinetHandler : Get a client handler to control EZSocketcore's client function

\*

\* Synopsis:

\* #include "EZSocketCore.h"

\* struct EZSocketCore \* GetClientHandler(struct Address\_and\_Port target , int \*errorcode);

\*

\* Description:

\* First parameter is the server address and port you want to connect.

\* struct Address\_and\_Port

\* {

\* char ip[IP\_MAX\_LENGTH];

\* int port;

\* };

\* Example:

\* struct Address\_and\_Port server;

\* memset(&server,0x0,sizeof(struct Address\_and\_Port));

\* strcpy(server.ip , "127.0.0.1");

\* server.port = 8080;

\*

\* You should pass a pointer which point to an integer to second parameter , when error occur,

\* GetClientHandler set errorcode end return NULL.

\*

\* --------------------------------------------------------------------------------------------------

\* Name:

\* GetClientErrorMsg : Get the error message when GetClientHandler return NULL

\*

\* Synopsis:

\* #include "EZSocketCore.h"

\* GetClientErrorMsg(int errorcode,char \*message,int maxlength)

\*

\* Description:

\* When GetClientHandler return NULL , you can call GetClientErrorMsg to get error message .

\* First parameter is the error code set by GetClientHandler , second parameter is a string pointer.

\* Third parameter is the max length of the sring.

\*

\* --------------------------------------------------------------------------------------------------

\* Name:

\* EZSocketCore.StartServerForever : Start the EZSocketCore server infinity loop

\*

\* Synopsis:

\* This function is c++ object-like member function in EZSocketCore structure

\* #include "EZSocketCore.h"

\* void (\*StartServerForever)( struct EZSocketCore \*pThis );

\*

\* Description:

\* EZSocketCore.StartServerForever nerver return after called .

\* It will wait clients forever.

\* When a client connection arrive, EZSocketCore api will fork and assign this client

\* to a sub process . It will automatic call the function defined by api user according to the parameter you

\* pass to GetServerHandler() .

\*

\* --------------------------------------------------------------------------------------------------

\* Name:

\* EZSocketCore.WriteToServer : write data to server

\*

\* Synopsis:

\* This function is c++ object-like member function write in EZSocketCore structure

\* int (\*WriteToServer)( struct EZSocketCore \*pThis , char \*buffer , int length );

\* Description:

\* first parameter : EZSocketCore structure

\* second parameter : buffer stores the message that will send to server

\* third parameter : length of second parameter

\* EZSocketCore.WriteToServer return how many byte actually send to server

\*

\* Example:

\* struct Address\_and\_Port server;

\* memset(&server,0x0,sizeof(struct Address\_and\_Port));

\* trcpy(server.ip,"127.0.0.1");

\* server.port=9999;

\* struct EZSocketCore \* ClientHandler = GetClientHandler(server,&Errorcode);

\* char buffer[100];

\* memset(buffer,0x0,100);

\* strcpy(buffer,"helloworld\n");

\* ClientHandler->WriteToServer(ClientHandler,buffer,strlen(buffer));

\*

\* --------------------------------------------------------------------------------------------------

\* Name:

\* EZSocketCore.ReadFromServer : read data from server

\*

\* Synopsis:

\* This function is c++ object-like member function write in EZSocketCore structure

\* int (\*ReadFromServer)( struct EZSocketCore \*pThis , char \*buffer , int maxlength );

\*

\* Description:

\* first parameter : EZSocketCore structure

\* second parameter : buffer used to store the data from server

\* third parameter : the max length of second parameter

\* EZSocketCore.WriteToServer return how many byte actually read from server

\*

\* Example:

\* struct Address\_and\_Port server;

\* memset(&server,0x0,sizeof(struct Address\_and\_Port));

\* trcpy(server.ip,"127.0.0.1");

\* server.port=9999;

\* struct EZSocketCore \* ClientHandler = GetClientHandler(server,&Errorcode);

\* char buffer[100];

\* ClientHandler->ReadFromServer(ClientHandler,buffer,sizeof(buffer));

\*

\* --------------------------------------------------------------------------------------------------

\* Name:

\* EZSocketCore.DisconnectToServer : disconnect to server

\*

\* Synopsis:

\* This function is c++ object-like member function write in EZSocketCore structure

\* void(\*DisconnectToServer)( struct EZSocketCore \*pThis);

\*

\* Description:

\* first parameter : EZSocketCore structure

\*

\* Example:

\* struct Address\_and\_Port server;

\* memset(&server,0x0,sizeof(struct Address\_and\_Port));

\* trcpy(server.ip,"127.0.0.1");

\* server.port=9999;

\* struct EZSocketCore \* ClientHandler = GetClientHandler(server,&Errorcode);

\* ClientHandler->DisconnectToServer(ClientHandler);

\*

\* --------------------------------------------------------------------------------------------------